In the United States Patent and Trademark Office
Board of Patent Appeals and Interferences

Appeal Brief

In re the Application of:

Michael A. ROTHMAN and Vincent J. ZIMMER

Serial No. 10/823,895
Docket Number: P19009
Firm No. 0077.0103
Filed: April 13, 2004

DEFRAGMENTING OBJECTS IN A STORAGE MEDIUM

Submitted by:

Konrad, Raynes & Victor LLP
315 So. Beverly Dr., Ste. 210
Beverly Hills CA 90212
(310) 556-7983
(310) 556-7984 (fax)

CERTIFICATE UNDER 37 CFR 1.8:
I hereby certify that this correspondence is being transmitted through the USPTO EFS-Web system over the Internet to the U.S. Patent and Trademark Office on August 14, 2009.

/David Victor/
David W. Victor

TABLE OF CONTENTS

I. Real Party in Interest

The entire right, title and interest in this patent application are assigned to real party in interest Intel Corporation.

II. Related Appeals, Interferences, and Judicial Proceedings

There are no prior and pending appeals, judicial proceedings or interferences known to the appellant which may be related to, directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

III. Status of the Claims

Claims 1-6, 8-15, 17-28, and 30-34 are pending and have been rejected.

The final rejection of the claims in the Final Office Action dated December 12, 2008 ("Dec. 2008 FOA") is being appealed for all pending claims 1-34.

Claims 7, 16, and 29 are canceled.

IV. Status of Amendments

Applicants submitted an amendment on August 13, 2009 following the filing of the Notice of Appeal and prior to the date of filing this Appeal Brief to cancel claims 7, 16, and 29. This Amendment has not yet been entered.

V. Summary of the Claimed Subject Matter

A. Independent Claim 1

The preamble recites a method. The Specification discloses that the embodiments may be implemented as a method. See, para. 19.

Below is an explanation of the claimed subject matter of claim 1 referring to the specification and drawings, where the claim requirements are underlined:

receiving an I/O request to write an update to an object in storage

With respect to this limitation, the Specification discloses receiving an I/O request to update a target object in the storage medium. See, FIG. 2, block 100 and paras. 11 and 16 of the Specification.

defragmenting the object in storage so that blocks in storage including the object are contiguous in response to receiving the I/O request to write the update to the object, wherein the request to write the update to the object causes the defragmentation operation

With respect to this limitation, the Specification discloses defragmenting the object in storage 14 so that blocks in storage 14 including the object are contiguous in response to receiving the I/O request. The I/O request may be to update the object. See, blocks 100, 112, FIG. 2 and paras. 15-16 of the Specification.

executing the I/O request to write the update to the object in storage

With respect to this limitation, the Specification discloses that the I/O request is executed with respect to the object in storage, and that the request may be to update the object. See, blocks 100, 112, FIG. 2, and paras. 15-16 of the Specification.

B. Independent Claim 10

The preamble recites a system in communication with storage. The Specification discloses a host 2 connected to a storage device 8. See, FIG. 1, paras. 7 and 8.

Below is an explanation of the claimed subject matter of claim 10 referring to the specification and drawings, where the claim requirements are underlined:

circuitry

The Specification discloses that the described operations may be performed by circuitry, where circuitry refers to either hardware or software or a combination thereof. See, para. 20. FIGs. 1 and 3 also show circuitry to perform the claimed defragmentation operations. See, Specification, paras. 9-10 and 17.

The limitations performed by the circuitry are disclosed in the Specification as discussed below.

receive an I/O request to write an update to an object in the storage

With respect to this limitation, the Specification discloses receiving an I/O request to update a target object in the storage medium. See, FIG. 2, block 100 and paras. 11 and 16 of the Specification.

defragment the object in storage so that blocks in storage including the object are contiguous in response to receiving the I/O request to write the update to the object, wherein the request to write the update to the object causes the defragmentation operation

With respect to this limitation, the Specification discloses defragmenting the object in storage 14 so that blocks in storage 14 including the object are contiguous in response to receiving the I/O request. The I/O request may be to update the object. See, blocks 100, 112, FIG. 2 and paras. 15-16 of the Specification.

execute the I/O request to write the update to the object in storage

With respect to this limitation, the Specification discloses that the I/O request is executed with respect to the object in storage, and that the request may be to update the object. See, blocks 100, 112, FIG. 2, and paras. 15-16 of the Specification.

C. Independent Claim 19

The preamble recites a system. FIG. 1 of the Specification discloses a system.

Below is an explanation of the claimed subject matter of claim 19 referring to the specification and drawings, where the claim requirements are underlined:

storage

The Specification discloses storage device 8. See, FIG. 1, paras. 9-10.

a storage controller coupled to the storage, wherein the storage controller is enabled to

The Specification discloses a storage controller 8 that performs the claimed operations, as described below. See, FIG. 1, paras. 9, 10, and 11.

The limitations performed by the storage controller are disclosed in the Specification as discussed below. The Specification discloses that the I/O logic 16 and defragmenter logic 18 of the storage controller 8 may implement the operations of FIG. 2. See, para. 11.

receive an I/O request to write an update to an object in the storage

With respect to this limitation, the Specification discloses receiving an I/O request to update a target object in the storage medium. See, FIG. 2, block 100 and paras. 11 and 16 of the Specification.

defragment the object in storage so that blocks in storage including the object are contiguous in response to receiving the I/O request to write the update to the object, wherein the request to write the update to the object causes the defragmentation operation

With respect to this limitation, the Specification discloses defragmenting the object in storage 14 so that blocks in storage 14 including the object are contiguous in response to receiving the I/O request. The I/O request may be to update the object. See, blocks 100, 112, FIG. 2 and paras. 15-16 of the Specification.

execute the I/O request to write the update to the object in storage

With respect to this limitation, the Specification discloses that the I/O request is executed with respect to the object in storage, and that the request may be to update the object. See, blocks 100, 112, FIG. 2, and paras. 15-16 of the Specification.

D. Independent Claim 23

The preamble recites an article of manufacture comprising at least one of a computer readable storage medium having code executed by a processor and a hardware device having logic to communicate with a storage and perform operations. The Specification discloses an article of manufacture these requirements in paras. 19-20.

Below is an explanation of the claimed subject matter of claim 23 referring to the specification and drawings, where the claim requirements are underlined:

receive an I/O request to write an update to an object in the storage

With respect to this limitation, the Specification discloses receiving an I/O request to update a target object in the storage medium. See, FIG. 2, block 100 and paras. 11 and 16 of the Specification.

defragment the object in storage so that blocks in storage including the object are contiguous in response to receiving the I/O request to write the update to the object, wherein the request to write the update to the object causes the defragmentation operation

With respect to this limitation, the Specification discloses defragmenting the object in storage 14 so that blocks in storage 14 including the object are contiguous in response to receiving the I/O request. The I/O request may be to update the object. See, blocks 100, 112, FIG. 2 and paras. 15-16 of the Specification.

execute the I/O request to write the update to the object in storage

With respect to this limitation, the Specification discloses that the I/O request is executed with respect to the object in storage, and that the request may be to update the object. See, blocks 100, 112, FIG. 2, and paras. 15-16 of the Specification.

VI. Grounds of Rejection to Be Reviewed on Appeal

A concise statement listing each ground of rejection presented for review is as follows:

A. Claims 1, 2, 8, 9, 10, 11, 17, 18, 19, 21, 23, 24, 30, and 31are rejected under (35 U.S.C. §103(a)) as obvious (35 U.S.C. §103) over Lawrence (U.S. Patent No. 6,253,300) in view of Andrew (U.S. Patent App. No. 2004/0059863).

B. Claims 3, 12, 20, and 25 are rejected under (35 U.S.C. §103) as obvious over Lawrence in view of Andrew and Brown (U.S. Patent No. 6,038,636).

C. Claim 22 is rejected under (35 U.S.C. §103) as obvious (35 U.S.C. §103(a)) over Lawrence, Andrew and further in view of Karger (U.S. Patent No. 5,339,449).

D. Claims 4, 5, 13, 14, 26, and 27 are rejected as obvious over Lawrence in view of Andrew and Douglis (U.S. Patent Pub. No. 2005/018075).

E. Claims 6, 15, and 28 are rejected as obvious (35 U.S.C. §103) over Lawrence in view of Andrew and Ball (U.S. Patent Pub. No. 2005/0162944).

F. Claims 32-34 are rejected as obvious (35 U.S.C. §103) over Lawrence in view of Andrew, Brown and Ball.

VII. Argument

A. Rejection Under 35 U.S.C. §103 Over Lawrence in view of Andrew

1. Claim 1, 10, 19, 21, and 23

Claims 1, 10, 19, and 23 require: receiving an I/O request to write an update to an object in storage; defragmenting the object in storage so that blocks in storage including the object are contiguous in response to receiving the I/O request to write the update to the object, wherein the request to write the update to the object causes the defragmentation operation; and executing the I/O request to write the update to the object in storage.

Applicants request review and reversal of the Examiner finding that col. 5, lines 37-42 of Lawrence teaches the claim requirement of defragmenting the object in storage so that blocks in storage including the object are contiguous in response to receiving the I/O request. (Dec. 2008 FOA, pgs. 2-3)

The cited col. 5 of Lawrence mentions that each file is stored in several locations separated by regions of the storage medium that do not hold the file's contents and that fragmentation can be alleviated or eliminated by running a defragmentation program on the files before copying them.

The cited col. 5 discussion of defragmenting before copying files does not disclose the claim requirement that an I/O request to write an update to the object causes defragmentation of the object. Although a defragmentation program may run before copying data, the cited col. 5 still does not teach defragmenting an object in response to receiving an I/O request to write an update to the object to which the defragmentation is directed.

The Examiner cited FIG. 4 of Andrew as triggering a defragmentation in response to an I/O request because a shutdown operation inherently comprises disk write operations. (Dec. 2008, FOA, pg. 3)

The cited FIG. 4 mentions that a hard drive is defragmented as part of a shutdown operation after temporary Internet files are removed. (Andrews, para. 35). However, the claims require that an object subject to an I/O operation is defragmented in response to an I/O request to update that object. The cited Andrews defragments a hard drive as

part of a shut down operation. Applicants submit that defragmenting a hard drive when performing a shutdown does not teach or suggest defragmenting an object in response to an update to that same object.

In the Response to Arguments, the Examiner found that Andrews teaching performing a defragmentation in response to an update to an object because Andrews defragmentation in response to a shutdown request comprises a write operations such as saving the system state. (Dec. 2008 FOA, pg. 11)

Applicants traverse this finding because in Andrews it is not the write to the system state information that triggers the defragmentation of the system as a whole in Andrews as the Examiner suggests, but a shutdown operation. Moreover, the claims require defragmenting a specific object in response to an update to the object. Andrews, on the other hand, defragments an entire hard drive as part of a shutdown, which involves writing system state information. In Andrews it is not writing system state information that triggers defragmentation, but the shutdown operation. Moreover, the writing of system state information does not trigger defragmentation of that system state information.

Thus, even if one combines Lawrence and Andrews as the Examiner proposes, the combination yields defragmenting files before copying the files (Lawrence) and defragmenting a hard drive as part of a shutdown operation (Andrews). This combination does teach or suggest that an object subject to an I/O operation is defragmented in response to an I/O request to update that object.

Accordingly, Applicants request that the Board reverse the rejection of claims 1, 10, 19, and 23 as obvious over the cited art because the requirements of these claims are not taught or suggested in the cited combination of Lawrence and Andrews.

Claim 21 is patentable over the cited art because it depends from claim 19, which is patentable over the cited art for the reasons discussed above.

2. Claims 2, 11, and 24

Claims 2, 11, and 24 depend from claims 1, 10, and 23, respectively, and further require that the I/O request is executed with respect to the object after defragmenting the object.

Applicants request review and reversal of the Examiner's finding that col. 5, lines 37-39 of Lawrence teaches the requirements of these claims. (Dec. 2008 FOA, pg. 4) Applicants traverse.

The cited col. 5 mentions defragmenting files before copying them. This does not teach or suggest updating the object after defragmenting the object.

In the Response to Arguments, the Examiner found that there is no patentable distinction between the cited defragmenting files before copying (Lawrence) and the claim requirement of executing the I/O request to update the object after defragmenting that object. The Examiner further found no patentable distinction between performing a defragmentation of a disk as part of a shutdown and the claim requirement of executing the I/O request to update the object after defragmenting that object. (Dec. 2008 FOA, pg. 12) Applicants submit that defragmenting an object after updating the object is different and not taught by defragmenting a file before copying a file or defragmenting a hard disk as part of a shutdown operation.

Accordingly, Applicants request that the Board reverse the rejection of claims 2, 11, and 24 as obvious over the cited art because these claims depend from base claims 1, 10, and 23, which are patentable over the cited art for the reasons discussed above, and because the additional requirements of these claims are not taught or suggested in the cited combination of Lawrence and Andrews.

3. Claims 8, 17, and 30

Claims 8, 17, and 30 depend from claims 1, 10, and 23 and further require operations of receiving the I/O request, initiating the operation to defragment the object, and executing the I/O request of defragmenting the object in storage are performed by a storage controller managing I/O requests to the storage.

Applicants request review and reversal of the Examiner's finding that Lawrence discloses these claim requirements because the defragmentation occurs in a computer and the computer inherently includes a storage controller and device driver. (Dec. 2008 FOA, pgs. 4-5)

Applicants traverse this finding because there is nothing inherent that defragmentation be initiated by the storage controller as opposed to some other computer

component. According to the Manual of Patent Examination and Procedure (MPEP), the "fact that a certain result or characteristic may occur or be present in the prior art is not sufficient to establish the inherency of that result or characteristic." MPEP Sec. 2112, pg. 47. Thus, the fact that defragmentation "may" be initiated in the storage controller as opposed to a program in the computer makes this finding of inherency inappropriate.

Applicants submit that although computers may have a storage controller as the Examiner notes, the Examiner has not cited any art that discloses, teaches or suggests that performing defragmenting of an object in response to a write to the object is performed by the storage controller. The Examiner is using hindsight to propose a modification to known computer components, such as a storage controller, that is not taught or suggested in the cited art.

Accordingly, Applicants request review and reversal of the Examiner's rejection of claims 8, 17, and 30 as obvious over the cited art because these claims depend from base claims 1, 10, and 23, which are patentable over the cited art for the reasons discussed above, and because the additional requirements of these claims are not taught or suggested in the cited combination of Lawrence and Andrews.

4. Claims 9, 18, and 31

Claims 9, 18, and 31 depend from claims 1, 10, and 23 and further require that the operation of defragmenting the object in storage is performed by a device driver for the storage providing an interface to the storage.

Applicants request review and reversal of the rejection of the Examiner finding that these claims are not patentably distinct because the Examiner has not shown where the cited Lawrence or Andrews teaches that defragmentation is performed by a device driver for the storage providing an interface to the storage as opposed to some other software program, such as an application program or utility. Thus, it is not inherent that a device driver perform the defragmentation.

Applicants request review and reversal of the Examiner's rejection of claims 9, 18, and 31 as obvious over the cited art because these claims depend from base claims 1, 10, and 23, which are patentable over the cited art for the reasons discussed above, and

because the additional requirements of these claims are not taught or suggested in the cited combination of Lawrence and Andrews.

B. Rejection Under 35 U.S.C. §103 Over Lawrence in view of Andrew and Brown

Claims 3, 12, 20, 25, and 32-34 a are patentable over the cited art because they depend from one of claims 1, 10, 19, and 23, respectively, which are patentable over the cited art for the reasons discussed above, and the additional requirements of these claims in combination with the base claims provide further grounds of patentability.

. Moreover, these claims provide additional grounds of patentability over the cited art for the following reasons.

1. Claim 3, 12, 20, and 25

Claims 3, 12, 20, and 25 depend from claims 1, 10, 19, and 23, respectively, and further require determining whether an amount of fragmentation of the object in the storage exceeds a fragmentation threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving the I/O request, wherein the object is defragmented if the amount of fragmentation exceeds the fragmentation threshold, and wherein the I/O request to update the object is executed without defragmenting the object in response to determining that the amount of fragmentation does not exceed the fragmentation threshold.

Applicants request review and reversal of the Examiner's finding that col. 5, lines 37-39 of Lawrence and col. 10, lines 1-5 and col. 7, lines 45-46 of Brown teach the additional requirements of these claims (Dec. 2008 FOA, pgs. 5-6)

The cited col. 5 of Lawrence mentions that fragmentation can be eliminated or alleviated by running a defragmentation program on the files before copying them. Applicants submit that this cited col. 5 does not teach or suggest determining whether an amount of fragmentation of an object exceeds a threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving a request to write update an object. Instead, the cited col. 5 mentions that one may run the defragmentation program to alleviate or eliminate fragmentation before copying files.

The cited col. 7 of Brown mentions that a file header includes a number indicating that the memory is valid, the name of a file, and a pointer to the next file, a number indicating the size of the file. The cited col. 10 mentions how to determine whether a file is contiguous by determining whether the size field in the header equals a predetermined code. Although the cited Brown mentions a number indicating a size of a file and using the size field in the header to determine whether the file is contiguous, this cited Brown does not teach or suggest determining whether an amount of fragmentation of an object exceeds a threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving a request to write update an object. Applicants submit a number indicating a size of a file used to determine whether a file is contiguous does not teach or suggest a threshold indicating a number of acceptable bytes of fragmentation.

In the Response to Arguments, the Examiner found that Andrews discussion of defragmenting in response to a shutdown involves defragmenting if disk defragmentation exists based on an interpretation of a threshold of zero and an "amount of fragmentation" exceeding zero. (Dec. 2008 FOA, pg. 12)

Applicants traverse this finding because the cited Andrews has no need for comparing an amount of defragmentation with thresholds to determine whether to perform the defragmentation because in Andrews defragmentation is triggered by a shutdown, so would be performed regardless of thresholds.

Moreover, the Examiner proposes an interpretation of the claim element "threshold" that would render the limitation meaningless by setting the threshold and amount of fragmentation to zero. Applicants submit such an interpretation of the "threshold" and "amount of fragmentation" should not be construed to have no limiting effect.

Applicants request review and reversal of the Examiner's rejection of claims 3, 12, 20, and 25 as obvious over the cited art because these claims depend from base claims 1, 10, and 23, which are patentable over the cited art for the reasons discussed above, and because the additional requirements of these claims are not taught or suggested in the cited combination of Lawrence, Andrews, and Brown.

C. Rejection Under 35 U.S.C. §103 Over Lawrence in view of Andrew and Karger

1. Claim 22

Claim 22 is patentable over the cited art because it depends from claim 19, which is patentable over the cited art for the reasons discussed above, and because the additional requirements of claim 22 in combination with the base claim provides further grounds of patentability.

D. Rejection Under 35 U.S.C. §103 Over Lawrence in view of Andrew and Douglis

Claims 4, 5, 13, 14, 26, and 27 a are patentable over the cited art because they depend from one of claims 1, 10, and 23, respectively, which are patentable over the cited art for the reasons discussed above, and because the additional requirements of these claims in combination with the base claims provide further grounds of patentability.

Moreover, these claims provide additional grounds of patentability over the cited art for the following reasons.

1. Claim 4, 5, 13, 14, 26, and 27

Claims 4, 13, and 26 depend from claims 1, 10, and 23, respectively, and further require determining whether a user settable flag indicates to perform defragmentation in response to receiving the I/O request, wherein the object is defragmented if the flag indicates to perform defragmentation.

The Examiner cited para. [0032] of Douglis as teaching the additional requirements of these claims. (Dec. 2008 FOA, pg. 8)

The cited para. [0032] discusses a power-aware monitor that monitors applications to defer execution of non-critical background tasks, that may be daemons or other application and whose execution is desirable only when there is not a restriction on power usage. Examples include full disk virus scans and defragmentation, among others.

Although the cited para. [0032] discusses a power monitor deferring defragmentation to execute when there is no restriction on power usage, nowhere does the cited para. [0032] anywhere teach or suggest a user settable flag that indicates to

perform defragmentation in response to receiving an update the object. Instead, the cited para. [0032] discusses deferring defragmentation for power management concerns, not indicating whether to perform a defragmentation in response to an update request as claimed.

In the Response to Arguments, the Examiner found that an indication of unrestricted power is a flag indicating to allow defragmentation because the flag is user settable because the user can alternate between restricted power and unlimited power. (Dec. 2008 FOA, pg. 13)

Applicants traverse this finding because although the state of unrestricted versus restricted power in Douglis may control whether defragmentation is performed, this does not teach a flag that indicates whether to perform defragmentation in response to receiving an I/O request to update an object.

Accordingly, Applicants request review and reversal of the Examiner's rejection of claims 4, 13, and 26 as obvious over the cited art because these claims depend from base claims 1, 10, and 23, which are patentable over the cited art for the reasons discussed above, and because the additional requirements of these claims are not taught or suggested in the cited combination of Lawrence, Andrews, and Douglis as discussed above.

Moreover, Applicants request reversal of the Examiner's rejection of claims 5, 14, and 27 because they depend from claims 4, 13, and 26, which are patentable over the cited art for the reasons discussed above.

E. Rejection Under 35 U.S.C. §103 Over Lawrence in view of Andrew and Ball

Claims 6, 15, and 28 a are patentable over the cited art because they depend from one of claims 1, 10, and 23, respectively, which are patentable over the cited art for the reasons discussed above, and because the additional requirements of these claims in combination with the base claims provide further grounds of patentability.

Moreover, these claims provide additional grounds of patentability over the cited art for the following reasons.

1. Claims 6, 15, and 28

Claims 6, 15, and 28 depend from claims 1, 10, and 23 and further require determining at least one logical partition including the object, wherein the object is defragmented if the object is within one logical partition and the I/O request to update the object is executed without defragmenting the object in response to determining that the object is included in more than one logical partition.

The Examiner cited the Abstract, the object 24, and para. 24 of Ball as teaching the additional requirements of these claims. (Dec., 2008 FOA, pg. 10) Applicants traverse.

The cited Abstract discuses a redundant memory architecture having an active memory and an inactive memory. The active memory supports in-service storage operations. The inactive memory is updated with stored contents of the active memory. Stored contents of the inactive memory are defragmented prior to an activity switch that results thenceforth in the inactive memory assuming the in-service storage operations and the active memory being updated with the stored contents of the inactive memory.

The cited para. [0024] of Ball mentions that the defragmentation can be performed on an inactive redundant memory, such that the in-service performance of a counterpart active memory need not be impacted.

Nowhere does the cited Ball teach or suggest defragmenting the object to update in response to determining that the object is included within one logical partition. Instead, the cited Abstract mentions that the inactive memory is defragmented prior to an activity switch that results in the inactive memory assuming the in-service storage operations and that the defragmentation can be performed on an inactive redundant memory.

In the Response to Arguments, the Examiner found that Andrews discloses defragmenting an object within a disk, which corresponds to one logical partition and that Bell teaches that portions of the object are not defragmented, wherein the object can be stored in multiple partitions.

Applicants traverse these findings because as discussed although Andrews may defragment an object within a disk, which could be a partition, there is no teaching that Andrews bases its defragmentation on whether the object is in one logical partition.

Instead, Andrews appears to perform its defragmentation without regard to whether the object is within one logical partition. Also, although Bell mentions that objects are stored in multiple partitions, as with Andrews, this factor does not control whether the defragmentation is in fact performed.

Accordingly, Applicants request review and reversal of the Examiner's rejection of claims 6, 15, and 28 as obvious over the cited art because these claims depend from base claims 1, 10, and 23, which are patentable over the cited art for the reasons discussed above, and because the additional requirements of these claims are not taught or suggested in the cited combination of Lawrence, Andrews, and Ball as discussed above.

F. Rejection Under 35 U.S.C. §103 Over Lawrence in view of Andrew, Brown, and Ball

1. Claim 32-34

Claims 32, 33, and 34 depend from claims 1, 10, and 23 and further require determining whether an amount of fragmentation of the object in the storage exceeds a fragmentation threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving the I/O request; determining at least one logical partition including the object, wherein the object is defragmented if the object is within one logical partition; and determining whether the object is read-only, wherein the object is defragmented if the object is not read-only, wherein the I/O request to update the object is executed without defragmenting the object in response to determining at least one of that the object is included in more than one logical partition, that the object is read-only, and that the amount of fragmentation does not exceed the fragmentation threshold.

The Examiner rejected claims 32-34 for the same reasons claims 3, 6, and 7 were rejected. (Dec. 2008 FOA, pg. 11) With respect to claim 3, 6, and 7 the Examiner cited col. 5, lines 37-39 of Lawrence, col. 10, lines 1-5 and col. 7, lines 45-46 of Brown, and the Abstract and element 24 of Ball. (Dec. 2008 FOA, pgs. 4, 6, 10

Applicants request review and reversal of the rejection of claims 32-34 because they depend from claims 1, 10, and 23 and because they include requirements claims 3, 6, and 7, all of which are patentable over the cited art for the reasons discussed above.

G. Conclusion

Each of the rejections set forth in the Final Office Action is improper and should be reversed.

Respectfully submitted,

/David Victor/

David W. Victor
Reg. No. 39,867
Konrad Raynes & Victor LLP
315 South Beverly Drive, Ste. 210
Beverly Hills, California 90212
Tel: 310-553-7977
Fax: 310-556-7984

Dated: August 14, 2009

VIII. Claims Appendix

1. (Previously Presented) A method, comprising:

receiving an I/O request to write an update to an object in storage;

defragmenting the object in storage so that blocks in storage including the object are contiguous in response to receiving the I/O request to write the update to the object, wherein the request to write the update to the object causes the defragmentation operation; and

executing the I/O request to write the update to the object in storage.

2. (Original) The method of claim 1, wherein the I/O request is executed with respect to the object after defragmenting the object.

3. (Previously Presented) The method of claim 1, further comprising:

determining whether an amount of fragmentation of the object in the storage exceeds a fragmentation threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving the I/O request, wherein the object is defragmented if the amount of fragmentation exceeds the fragmentation threshold, and wherein the I/O request to update the object is executed without defragmenting the object in response to determining that the amount of fragmentation does not exceed the fragmentation threshold.

4. (Original) The method of claim 1, further comprising:

determining whether a user settable flag indicates to perform defragmentation in response to receiving the I/O request, wherein the object is defragmented if the flag indicates to perform defragmentation.

5. (Original) The method of claim 4, further comprising:

executing the I/O request without performing defragmentation if the flag does not indicate to perform defragmentation.

6. (Previously Presented) The method of claim 1, further comprising:

determining at least one logical partition including the object, wherein the object is defragmented if the object is within one logical partition, wherein the I/O request to update the object is executed without defragmenting the object in response to determining that the object is included in more than one logical partition.

7. (Canceled)

8. (Previously Presented) The method of claim 1, wherein the operations of receiving the I/O request, initiating the operation to defragment the object, and executing the I/O request of defragmenting the object in storage are performed by a storage controller managing I/O requests to the storage.

9. (Original) The method of claim 1, wherein the operation of defragmenting the object in storage is performed by a device driver for the storage providing an interface to the storage.

10. (Previously Presented) A system in communication with storage, comprising:

circuitry enabled to:

receive an I/O request to write an update to an object in the storage;

defragment the object in storage so that blocks in storage including the object are contiguous in response to receiving the I/O request to write the update to the object, wherein the request to write the update to the object causes the defragmentation operation; and

execute the I/O request to write the update to the object in storage.

11. (Original) The system of claim 10, wherein the I/O request is executed with respect to the object after defragmenting the object.

12. (Previously Presented) The system of claim 10, wherein the circuitry is further enabled to:

determine whether an amount of fragmentation of the object in the storage exceeds a fragmentation threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving the I/O request, wherein the object is defragmented if the amount of fragmentation exceeds the fragmentation threshold, and wherein the I/O request to update the object is executed without defragmenting the object in response to determining that the amount of fragmentation does not exceed the fragmentation threshold.

13. (Original) The system of claim 10, wherein the circuitry is further enabled to: determine whether a user settable flag indicates to perform defragmentation in response to receiving the I/O request, wherein the object is defragmented if the flag indicates to perform defragmentation.

14. (Original) The system of claim 13, wherein the circuitry is further enabled to: execute the I/O request without performing defragmentation if the flag does not indicate to perform defragmentation.

15. (Previously Presented) The system of claim 10, wherein the circuitry is further enabled to:

determine at least one logical partition including the object, wherein the object is defragmented if the object is within one logical partition, wherein the I/O request to update the object is executed without defragmenting the object in response to determining that the object is included in more than one logical partition.

16. (Canceled)

17. (Previously Presented) The system of claim 10, wherein the circuitry is implemented in a storage controller managing I/O requests to the storage, wherein the operations of receiving the I/O request, initiating the operation to defragment the object,

and executing the I/O request of defragmenting the object in storage are performed by the storage controller.

18. (Original) The system of claim 10, wherein the circuitry is implemented in a device driver interfacing between an operating system and the storage, and wherein the operation of defragmenting the object in storage is performed by the device driver.

19. (Previously Presented) A system, comprising:

storage;

a storage controller coupled to the storage, wherein the storage controller is enabled to:

receive an I/O request to write an update to an object in the storage;

defragment the object in storage so that blocks in storage including the object are contiguous in response to receiving the I/O request to write the update to the object, wherein the request to write the update to the object causes the defragmentation operation; and

execute the I/O request to write the update the object in storage.

20. (Previously Presented) The system of claim 19, wherein the storage controller is further enabled to:

determine whether an amount of fragmentation of the object in the storage exceeds a fragmentation threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving the I/O request, wherein the object is defragmented if the amount of fragmentation exceeds the fragmentation threshold, and wherein the I/O request to update the object is executed without defragmenting the object in response to determining that the amount of fragmentation does not exceed the fragmentation threshold.

21. (Original) The system of claim 19, wherein the storage controller and storage device are included in a same housing.

22. (Original) The system of claim 19, further comprising:

a processor; and

a memory enabled to store the I/O request before the I/O request is received by the storage controller.

23. (Previously Presented) An article of manufacture comprising at least one of a computer readable storage medium having code executed by a processor and a hardware device having logic to communicate with a storage and perform operations, the operations comprising:

receive an I/O request to write an update to an object in storage;

defragment the object in storage so that blocks in storage including the object are contiguous in response to receiving the I/O request to write the update to the object, wherein the request to write the update to the object causes the defragmentation operation; and

execute the I/O request to write the update to the object in storage.

24. (Original) The article of manufacture of claim 23, wherein the I/O request is executed with respect to the object after defragmenting the object.

25. (Previously Presented) The article of manufacture of claim 23 further enabled to:

determine whether an amount of fragmentation of the object in the storage exceeds a fragmentation threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving the I/O request, wherein the object is defragmented if the amount of fragmentation exceeds the fragmentation threshold, and wherein the I/O request to update the object is executed without defragmenting the object in response to determining that the amount of fragmentation does not exceed the fragmentation threshold.

26. (Original) The article of manufacture of claim 23 further enabled to:

determine whether a user settable flag indicates to perform defragmentation in response to receiving the I/O request, wherein the object is defragmented if the flag indicates to perform defragmentation.

27. (Previously Presented) The article of manufacture of claim 26 further enabled to:

execute the I/O request without performing defragmentation if the flag does not indicate to perform defragmentation.

28. (Previously Presented) The article of manufacture of claim 23 further enabled to:

determine at least one logical partition including the object, wherein the object is defragmented if the object is within one logical partition, wherein the I/O request to update the object is executed without defragmenting the object in response to determining that the object is included in more than one logical partition.

29. (Canceled)

30. (Original) The article of manufacture of claim 23 wherein the operation of defragmenting the object in storage is performed by a storage controller managing I/O requests to the storage.

31. (Previously Presented) The article of manufacture of claim 23, wherein the operations of receiving the I/O request, initiating the operation to defragment the object, and executing the I/O request of defragmenting the object in storage are performed by a device driver for the storage providing an interface to the storage.

32. (Previously Presented) The method of claim 1, further comprising:

determining whether an amount of fragmentation of the object in the storage exceeds a fragmentation threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving the I/O request;

determining at least one logical partition including the object, wherein the object is defragmented if the object is within one logical partition; and

determining whether the object is read-only, wherein the object is defragmented if the object is not read-only, wherein the I/O request to update the object is executed without defragmenting the object in response to determining at least one of that the object is included in more than one logical partition, that the object is read-only, and that the amount of fragmentation does not exceed the fragmentation threshold.

33. (Previously Presented) The system of claim 10, wherein the circuitry is further enabled to:

determine whether an amount of fragmentation of the object in the storage exceeds a fragmentation threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving the I/O request;

determine at least one logical partition including the object, wherein the object is defragmented if the object is within one logical partition; and

determine whether the object is read-only, wherein the object is defragmented if the object is not read-only, wherein the I/O request to update the object is executed without defragmenting the object in response to determining at least one of that the object is included in more than one logical partition, that the object is read-only, and that the amount of fragmentation does not exceed the fragmentation threshold.

34. (Previously Presented) The article of manufacture of claim 23, further comprising:

determining whether an amount of fragmentation of the object in the storage exceeds a fragmentation threshold indicating an acceptable number of bytes stored in non-contiguous locations in response to receiving the I/O request;

determining at least one logical partition including the object, wherein the object is defragmented if the object is within one logical partition; and

determining whether the object is read-only, wherein the object is defragmented if the object is not read-only, wherein the I/O request to update the object is executed without defragmenting the object in response to determining at least one of that the object is included in more than one logical partition, that the object is read-only, and that the amount of fragmentation does not exceed the fragmentation threshold.

IX. Evidence Appendix

None

X. Related Proceedings Appendix

None